Pricing Options under Jump-Diffusion Models by Adaptive Radial Basic Functions

Ron Chan

No. 06/10

BATH ECONOMICS RESEARCH PAPERS

Department of Economics

Pricing Options under Jump-diffusion Models by Adaptive Radial Basis Functions

Ron Chan*
School of Economics, Mathematics & Statistics
Birkbeck,
University of London
Malet Street, London, WC1X 9JX, UK,
Version 1.5

June 7, 2010

Abstract

The aim of this paper is to show that option prices in jump-diffusion models can be computed using meshless methods based on Radial Basis Function (RBF) interpolation instead of traditional mesh-based methods like Finite Differences (FDM) or Finite Elements (FEM). The RBF technique is demonstrated by solving the partial integro-differential equation for American and European options on non-dividend-paying stocks in the Merton jump-diffusion model, using the Inverse Multiquadric Radial Basis Function (IMQ). The method can in principle be extended to Lévy-models. Moreover, an adaptive method is proposed to tackle the accuracy problem caused by a singularity in the initial condition so that the accuracy in option pricing in particular for small time to maturity can be improved.

Keywords: adaptive method, Lévy processes, option pricing, parabolic partial integro-differential equations, singularity, Radial Basis Function, the Merton Jump-diffusions Model.

Acknowledgements: I would like to express my gratitude to Raymond Brummelhuis and Simon Hubbert, School of Economics, Mathematics and Statistics, Birkbeck, University of London; and Shingyu Leung, Department of Mathematics, University of California, Irvine, for their comments and advice on this and earlier versions of the paper.

*Email: rchan@ems.bbk.ac.uk

1 Introduction

In this paper we show how to compute European and American option prices in the Merton jump-diffusion model using Radial Basis Function (RBF) interpolation techniques. RBF methods have recently been proposed for numerically solving initial value and free boundary problems for the classical Black and Scholes equation, both in the one and in the multiple asset case [10, 12, 14]. The new feature of the present paper is that in the Merton model (and comparable jump-diffusion models, as in general Lévy type models), the Black and Scholes PDE is replaced by a Partial Integro-Differential Operator or PIDE, involving a non-local term in the form of an integral operator. Our main contribution is to show how to numerically solve these in an efficient way using RBFs, both for initial value and free boundary problems (as for American options), and including when singularities in the initial value (identified with the option's pay-off) are present. We have chosen the Merton jump-diffusion model as a typical case on which to test the present RBF methodology. Our method extends however without problems to other contexts in which the basic pricing equation is a PIDE, like that of Lévy-type models such as CGMY [4] or Variance Gamma [21]. These will be treated in a future paper.

Currently, PIDEs such as the Merton one have mostly been treated by a traditional Finite Difference Method (FDM), or by a Finite Element Method (FEM).The idea is to simply fully discretize the PIDE on an equidistant grid, after having (artificially) localized the equations to some bounded interval/domain in $\mathbb{R}$. The non-local integral term can be computed by numerical quadrature or by using the Fast Fourier Transform (FFT). In general, there are a number of problems which arise with these current approaches:

- The option price behaviour outside the solution domain must be assumed; see e.g. [5, 7, 8].
- Some of the literature, e.g. [1, 2, 5, 22], has played down the importance of pricing American and European vanilla option values when time to maturity is less than six months. The reason is that for short times-to-maturity the numerical methods used price the option incorrectly around the strike price where a singularity (kink) exists. A singularity is defined as a point at which the function, or its derivative, is discontinuous. The payoff functions of vanilla call and put options have such a singularity. As a result, standard numerical methods like FDM and FEM cannot give accurate precision and suffer a reduced rate of convergence when one uses them to price options at a very short time to maturity. Foysth *et al.* shed light on addressing this kind of problem [7] by suggesting Rannacher's time stepping method [26]. This is a mixture of implicit and Crank-Nicolson methods. They demonstrate this technique by approximating an option price whose maturity is a quarter of a year. This method gives second order rates of convergence when pricing European options but not for American ones. By using the same idea and combining it with a penalty method and a modified form of a timestep selector suggested in [16], Forysth *et al.* in their other paper [8] show how to achieve second order convergence for pricing American options. Although their methods can yield second order convergence, the necessary calculations can be quite complex.
- In [1, 2, 7, 8], etc, the Fast Fourier Transform is applied to calculate the non-local jump integral term in the PIDE, and the diffusion and integral terms are treated separately. This therefore requires that function values are interpolated and extrapolated between the diffusion and integral grids so as to approximate the convolution term.
- Andersen and Andreasen's approach of combining an operator splitting approach with the Fast Fourier Transform (FFT) approximation of a convolution integral to price European options with jump diffusion [1] cannot deal easily with American options.
- The papers [2, 5, 7, 8] implement an implicit-explicit numerical scheme to price European or American options under the Merton jump-diffusion model. These papers treat the convection (hyperbolic) term of the PIDE explicitly by implementing the upwind scheme and the diffusion (elliptic) term of the PIDE implicitly. As a result, restrictive stability conditions are necessary for the convection term when the upwind scheme is implemented.

- A final but fundamental problem with both FDM and FEM is that these are, in practice, restricted to problems of two or three space dimensions; however, most applications easily need many more, e.g. when pricing basket options.

Our RBF-method will circumvent many of these disadvantages. In particular, differential and integral terms will be treated on an equal footing, and the use of an adaptive RBF-scheme will allow us to deal with the singularity in the option pay-off. This paper is divided into five sections, including this introduction. Section 2 is a brief review of Metron's jump-diffusion model. In section 3 we first explain adaptive residual subsampling method and then define our RBF algorithm for solving PIDEs, which we implement the Merton PIDE. Section 4 contains our numerical results for interpolation of an initial put payoff function and for both European and American put options, including an analysis of the max error, the root-mean-square error and the relative error. Section 5 concludes.

2 European and American options in a Merton jump-diffusion Market

In this paper we focus our attention on the classical Merton jump-diffusion model with Gaussian jumps [23]. This model can be considered as a particular example of a Lévy model for describing the price dynamics of the underlying risky asset, $(S_t)_{t\geq 0}$, in a financial market. The evolution of $(S_t)_{t\geq 0}$ is driven by a diffusion process, punctuated by jumps which describe rare events such as crashes and drawdowns at random intervals. As a market model, it is an example of an incomplete market. We will skirt around the hedging issue by working directly in the risk-neutral probability measure $\mathbb{Q}$, as is customary. The stock price process,$(S_t)_{t\geq 0}$, is then given by

$$S_t = S_0 e^{rt+X_t} \tag{1}$$

where S_0 is the stock price at time zero, r is the risk-free interest rate and X_t is defined by:

$$X_t := (-\lambda\eta - \frac{\sigma^2}{2})t + \sigma W_t + \sum_{i=1}^{N_t} Y_i, \tag{2}$$

W_t is a Brownian motion, N_t is a Possion process with intensity λ, Y_i is an iid sequence of normally distributed $\mathbb{N}(\mu_j, \sigma_j^2)$ variables, and $\eta := \mathbb{E}(e^{X_t} - 1) := e^{\mu_j + \sigma_j^2/2} - 1$, the expected relative price change due to a jump. The drift-term in (1) assumes that $e^{-rt}S_t$ is a martingale with respect to the natural filtration. We let $\tau = T - t$, the time-to-maturity, where T is the maturity of the financial option under consideration and we introduce $x = \log S_t$, the underlying asset's log-price. If $u(x, \tau)$ denotes the values of some (American and European) contingent claim on S_t when $\log S_t = x$ and $\tau = T - t$, then it is well-known, see for example, [6] that u satisfies the following PIDE in the non-exercise region:

$$\frac{\partial u(\tau, x)}{\partial \tau} - \mathcal{L}u(x, \tau) + ru(x, \tau) = 0, \text{ in } (0, T) \times \mathbb{R} \tag{3}$$

where $\mathcal{L}$ is the infinitesimal generator of the transition semigroup of the driving Lévy process. Explicitly, $\mathcal{L}$ is given by:

$$\mathcal{L}u(x, \tau) = \frac{\sigma^2}{2}u_{xx} + \left(r - \frac{\sigma^2}{2} + \lambda\eta\right)u_x - \lambda u + \lambda \int_{-\infty}^{\infty} u(x + y, \tau) f(y) dy,$$

A European option can be exercised only at the expiry date (maturity) of the option, i.e. at a single predefined point in time. Consider a European put on the underlying non-dividend-paying $S(t) = e^{x_t}$, with maturity T, and strike K. In terms of logarithm price $x = \log S_t$, the pay-off at $t = T$ or $\tau = 0$ is:

$$u(x, 0) = H(e^x) = \max\{K - e^x, 0\} \tag{4}$$

and one can price this put option by solving (3) with initial condition (4).

For an American put, we have to take into account the possibility of early exercise, e.g. [6, 15, 24]. As a result, the highest value of American option can be achieved by maximizing over all allowed exercise strategies:

$$u(x,\tau) = \operatorname{ess\,sup}_{\tau^* \in \Gamma(t,T)} \mathrm{E}_t^{\mathrm{Q}} \left[\mathrm{e}^{-\mathrm{r}(\tau^* - t)} \mathrm{H}\left(\mathrm{e}^{\mathrm{x}_{\tau^*}}\right) \right] \tag{5}$$

where $\Gamma(t,T)$ denotes the set of non-anticipating exercise times τ^*, satisfying $t \leq \tau^* \leq T$. To actually compute the $u(x,\tau)$ of the American put, one can solve the following linear complementarity problem [6, 28]:

$$\frac{\partial u(\tau,x)}{\partial \tau} - \mathcal{L}u(x,\tau) + ru(x,\tau) \geq 0, \text{ in } (0,T) \times \mathbb{R} \tag{6}$$

$$u(x,\tau) - H(e^x) \geq 0, \text{ a.e. in } (0,\mathrm{T}) \times \mathbb{R} \tag{7}$$

$$\big(u(x,\tau) - H(e^x)\big) \left(\frac{\partial u(x,\tau)}{\partial \tau} - \mathcal{L}u(x,\tau) + ru(x,\tau) \right) = 0, \text{ in } (0,T) \times \mathbb{R} \tag{8}$$

$$u(x,0) = H(e^x), \tag{9}$$

Since we only deal with a jump-diffusion model with $\sigma > 0$ and finite jump intensity in this paper, we know that by Pham [24], the smooth pasting condition,

$$\frac{\partial u(x_{\tau^*}, \tau^*)}{\partial x} = -1$$

is valid at time of exercise $\tau*$. Therefore the value of an American put option is continuously differentiable with respect to the underlying on $(0,T) \times \mathbb{R}$; in particular the derivative is continuous across the exercise boundary.

3 Meshfree Numerical Approximation Method

Meshfree radial basis function (RBF) interpolation is a well-known technique for reconstructing an unknown function from scattered data. It has numerous applications in different fields, such as terrain modeling in geology, surface reconstruction in imaging, and the numerical solution of partial differential equations in applied mathematics. In particular, RBFs have recently been used to solve the PDEs of quantitative finance. A number of authors, including Fausshauer *et al.* [10, 12] and Hon and Mao [14], have suggested RBFs as a tool for solving Black-Scholes equations for European as well as American options. However, because of the non-smoothness of the payoffs of financial derivatives like calls and puts, the RBF methodology when naively implemented on an equidistant grid may not yield correct option prices for small times-to-maturity. In particular, numerically computed call and put prices may become negative near the strike, when close to maturity. To address this problem, we will use an adaptive RBF method. Recently, Sarra [27] and Driscoll and Heryudono [9] have suggested using adaptive RBF to solve a PDE whose solution curve has singularities. They have illustrated their techniques by solving a time dependent Burgers' equation. In brief, their idea is to refine the solution by putting more interpolation points of RBFs around or at the singularities so as to reduce the error of the RBF-approximation. We will use a similar method in section 3.1 below. The aim here is to obtain a very good RBF approximation of the initial value or pay-off of the option. Once we dispose of such a good-quality RBF-interpolant, we implement an RBF-scheme to solve the PIDE with this RBF-interpolant as initial value. The general idea of the proposed numerical scheme is to approximate the unknown function $u(x,\tau)$ by a RBF-interpolant using the interpolation points found for the initial value using the adaptive RBF-scheme, and derive a system of linear constant coefficient ODE by requiring that the PIDE (3) be satisfied in the chosen RBF-interpolation points.

A typical RBF in dimension n is a rotation-invariant function on $\mathbb{R}^n$, usually written as $\phi(||x||)$ with $||x||$ the Euclidean norm, and ϕ a suitable univariate function, such that for any set of N points $x_1, \ldots, x_N \in \mathbb{R}^n$ the matrix $\big(\phi(||x_i - x_j||)\big)_{1 \le i,j \le N}$ is non-singular. Such functions ϕ exist (see below), and the RBF-interpolant of a given function f on a given set of *interpolation points*, $x_1, \ldots, x_n$, is defined as $\sum_{j=1}^{N} \rho_j \phi(||x - x_j||)$, where the coefficients ρ_j are determined by

$$f(x_i) = \sum_{j=1}^{N} \rho_j \phi(||x_i - x_j||).$$

The non-singularity condition on ϕ implies the unique solvability of this system in $(\rho_1, \ldots, \rho_N)$. In applications, the data points $x_1, \ldots, x_N$ can be arbitrarily scattered in space and do not have to belong to some pre-existing mesh. Commonly used RBFs are :

$$\phi(r) = \begin{cases} \sqrt{(cr)^2 + 1} & \text{for Multiquadric (MQ),} \\ \frac{1}{\sqrt{(cr)^2+1}} & \text{for Inverse Multiquadric (IMQ),} \\ \exp(-c^2 r^2) & \text{for Gaussian,} \\ r^2 \log(r) & \text{for Thin Plate Spline (TPS).} \end{cases}$$

where c is called a *shape parameter*, a user defined parameter which can be fine-tuned to improve the accuracy of the RBF-approximation. In this paper we will use the IMQ. Also, since we only deal with the one dimensional case, we can simplify $\phi(||x - x_j||_2)$ to $\phi(|x - x_j|)$.

After picking interpolation points $x_j \in \mathbb{R}$, we approximate, for any fixed time-to-maturity τ, the solution $u(x, \tau)$ in (3) by its RBF-interpolant:

$$u(x, \tau) \simeq \sum_{j=1}^{N} \rho_j(\tau) \phi(||x - x_j||_2) =: \widehat{u}(x, \tau), \tag{10}$$

Since the radial basis function does not depend on time, the time derivative of $\widehat{u}(x, \tau)$ in equation (10) is simply:

$$\frac{\partial \widehat{u}(x, \tau)}{\partial \tau} = \sum_{j=1}^{N} \frac{d\rho_j(\tau)}{d\tau} \phi(|x - x_j|), \tag{11}$$

Moreover, the first and second partial derivatives of $\widehat{u}(x, \tau)$ with respect to x are

$$\frac{\partial \widehat{u}(x, \tau)}{\partial x} = \sum_{j=1}^{N} \rho_j(\tau) \frac{\partial \phi(|x - x_j|)}{\partial x}, \tag{12}$$

$$\frac{\partial^2 \widehat{u}(x, \tau)}{\partial x^2} = \sum_{j=1}^{N} \rho_j(\tau) \frac{\partial^2 \phi(|x - x_j|)}{\partial x^2}, \tag{13}$$

where for the particular case when ϕ is an IMQ,

$$\begin{aligned} \frac{\partial \phi(|x - x_j|)}{\partial x} &= -\frac{c^2(x - x_j)}{\Big(\big(c(x - x_j)\big)^2 + 1\Big)^{3/2}}, \\ \frac{\partial^2 \phi(|x - x_j|)}{\partial x^2} &= c^2 \frac{2\big(c(x - x_j)\big)^2 - 1}{\Big(\big(c(x - x_j)\big)^2 + 1\Big)^{5/2}}, \end{aligned} \tag{14}$$

3.1 Adaptive Residual Subsampling Method v.s. Equally Spacing Method

In this section, we describe two methods for choosing the interpolation points $x_1, \ldots, x_n \in \mathbb{R}$: the straightforward Equally Spacing Method (ESM) used in [10, 12, 14] and the more sophisticated adaptive residual subsampling method (ARSM) from [9].

In the ESM, we determine an interval $[x_{\min}, x_{\max}]$ outside of which we can neglect the contribution of $u(x,\tau)$ to the non-local integral term of an PIDE (3), and for given $N = 1, 2, \ldots$, simply put

$$x_j := x_j^{\Delta x} = x_{\min} + j\Delta x,\ j = 1, 2, \ldots, N \tag{15}$$

where $\Delta x = (x_{\max} - x_{\min})/N$,

For the ARSM, we start first to generate an initial N points of x_j using ESM and determine the RBF approximand of the initial function (4). We then compute the interpolation error at evaluation points halfway between initial interpolation points. Points at which the error exceeds a threshold of refinement, θ_r, become new interpolation points, and points that lie between two points whose error is below a threshold of coarseness, θ_c, are removed. We will specify θ_r and θ_c in the next section. The two end points are always left intact. The shape parameter, c, of each center is chosen based on the distance to the nearest neighbours (we will explain the choice of c in the next paragraph), and the RBF approximation of function (4) is recalculated using the new set of interpolation points after which the procedure is repeated. In brief, the adaptation process follows the familiar paradigm of solve-estimate-refine/coarsen until a stopping criterion is reached. For more details of this algorithm and matlab code, we refer the reader to [9].

In both methods, we choose an appropriate shape parameter in our IMQ so as to achieve a high degree of accuracy for our approximation of $u(x,\tau)$. There exits a substantial literature on choosing optimal shape parameters in IMQ or other types of RBFs, e.g. [11], [13] and [17]. Here we choose a shape parameter of $1/(4\Delta x)$, as proposed by Frasshauer *et al.* [10] and Hon *et al.* [14], where Δx is the distance between two neighboring nodes of interpolation points. For the ESM, Δx is of course constant but for the ARSM, it isn't. This may cause potential problems for the invertibility of the RBF-interpolation matrix. In the case of the ESM we have that with the interpolation points chosen according to (15), our RBF is $\phi_{\Delta x}(|x|)=\phi_{\Delta x}(x)=\left((\frac{x}{4\Delta x})^2+1\right)^{-\frac{1}{2}}$, and consequently:

$$\left(\phi_{\Delta x}(x_j^{\Delta x} - x_k^{\Delta x})\right)_{j,k} = \left(\frac{1}{\sqrt{(\frac{j-k}{4})^2+1}}\right)_{1\le j,k\le N}.$$

Positive definitness of this matrix is guaranteed by general RBF-theory, e.g. by the positivity of the Fourier transform of $\left((\frac{x}{4\Delta x})^2+1\right)^{-\frac{1}{2}}$, cf. Buhmann [3], Powell [25] or Wendland [30] . For the ARSM however, the matrix to be inverted is of the form

$$\left(\phi_{c_k}(x_j - x_k)\right)_{j,k},\ \text{where}\, c_k = \frac{1}{4\Delta x_k}, \tag{16}$$

with Δx_k, the nearest-neighbour distance to x_k, and in general is not symmetric. Invertibility is not guaranteed by general theory anymore, but has to be numerically checked at each stage, as part of the algorithm. This has not led to any problems in the implementation. The idea of using adaptive methods to gain high orders of accuracy in interpolation and numerical solution of PDEs has exploited in a number of papers by Kansa *et al.*, cf. [17, 18, 19, 20]. They use Multiquadric or MQ, and implement an adaptive shape parameter c_k of the form:

$$c_k = \frac{1}{c_{\min}}\left(\frac{c_{\min}}{c_{\max}}\right)^{\frac{k-1}{N-1}},\ k = 1, 2, \ldots, N,$$

where $c_{\min}$ and $c_{\max}$ are two input constant parameters. In [17], Kansa and Carlson compare the accuracy of interpolation of univariate and bivariate test functions using this adaptive shape parameter with that of

using a constant shape parameter. They conclude that there is a dramatic improvement in the interpolation errors by using variable shape parameters. In [18, 19, 20], they also show the effectiveness of using variable shape parameters to improve the accuracy of solving time-dependent PDEs in one and three dimensions by comparing the results with those obtained by using Finite Difference Methods.

3.2 Transforming PIDE to a system of ODEs by RBF

Given a set of interpolation points $x_1, \ldots, x_j, \ldots, x_N$, and a RBF ϕ, we can construct $N \times N$ matrices $\boldsymbol{A}$, $\boldsymbol{A}_x$ and $\boldsymbol{A}_{xx}$ defined by $\big(\phi(|x_i - x_j|)\big)_{1\le i,j\le N}$, $\big(\phi^{'}(|x_i - x_j|)\big)_{1\le i,j\le N}$ and $\big(\phi^{''}(|x_i - x_j|)\big)_{1\le i,j\le N}$ respectively. Note in case the x_j's are chosen according to the ESM (15), $\phi(x)$ actually depends itself on N (or Δx). We also define a matrix-valued function $y \to \boldsymbol{A}(y)$ by $\big(\phi(|x_i + y - x_j|)\big)_{1\le i,j\le N}$. If we substitute $\widehat{u}(x,\tau)$ for $u(x,\tau)$ in (3) and require the PIDE to be satisfied in the interpolation points x_j, we arrive at the following system of ODEs for the vector $\boldsymbol{\rho}(\tau) := \big(\rho_1(\tau), \ldots, \rho_N(\tau)\big)$

$$\boldsymbol{A}\boldsymbol{\rho}_\tau = \frac{\sigma^2}{2}\boldsymbol{A}_{xx}\boldsymbol{\rho} + \left(r - \frac{\sigma^2}{2} - \lambda\eta\right)\boldsymbol{A}_x\boldsymbol{\rho} + (r+\lambda)\boldsymbol{A}\boldsymbol{\rho} + \lambda\left(\int_{-\infty}^{\infty}\boldsymbol{A}(y)f(y)\,\mathrm{d}y\right)\boldsymbol{\rho}, \tag{17}$$

where $\rho_\tau := \frac{\partial\rho}{\partial\tau}$, and where we recall that $f(y)$ is the probability density of the jump $Y_i \sim \mathbb{N}(\mu_J, \sigma_J^2)$: $f(y) = (\sigma_J\sqrt{2\pi})^{-1}\exp\big(-(y-\mu_J)^2/2\sigma_J^2\big)$.

Before applying a suitable numerical integration algorithm to the integral terms in (17), we truncate the integrals from an infinite computational range into a finite one. Briani *et al.* [2], Cont and Voltchkova [5] and Forysth *et al.* [7, 8] have provided different numerical techniques to find out a finite computational range so as to reduce the numerical approximation errors when doing this truncation. Numerical experimentation has shown that for the model parameters considered in this paper we get good results by simply cutting off the integral at $5\sigma_J + \mu_j$. We therefore transform equation (17) into

$$\boldsymbol{A}\boldsymbol{\rho}_\tau = \frac{\sigma^2}{2}\boldsymbol{A}_{xx}\boldsymbol{\rho} + \left(r - \frac{\sigma^2}{2} - \lambda\eta\right)\boldsymbol{A}_x\boldsymbol{\rho} + (r+\lambda)\boldsymbol{A}\boldsymbol{\rho} + \lambda\left(\int_{-b}^{b}\boldsymbol{A}(y)f(y)\,\mathrm{d}y\right)\boldsymbol{\rho}. \tag{18}$$

where $b = 5\sigma_J + \mu_j$. We use matlab's vectorized quadrature to evaluate the matrix of the integrals in (18): this amounts to approximating

$$\int_{-b}^{b}\phi(|x_i + y - x_j|)f(y)\,\mathrm{d}y \approx \sum_{k=1}^{m} w_k\phi(|x_i + y_k - x_j|)f(y_k), \tag{19}$$

where w_k and y_k are suitable quadrature weights and quadrature points; cf. [29] for details. To simplify notations, we set

$$F(x_i - x_j) = \sum_{k=1}^{m} w_k\phi(|x_i + y_k - x_j|)f(y_k).$$

Then the integrals in equation (18) will be approximated by

$$\begin{aligned}\int_{-b}^{b}\boldsymbol{A}(y)f(y)\,\mathrm{d}y &\approx \begin{bmatrix} F(x_1-x_1) & F(x_1-x_2) & \ldots & F(x_1-x_N) \\ F(x_2-x_1) & F(x_2-x_2) & \ldots & F(x_2-x_N) \\ \ldots & \ldots & \ldots & \ldots \\ F(x_N-x_1) & F(x_N-x_2) & \ldots & F(x_N-x_N) \end{bmatrix} \\ &= \boldsymbol{C}(y).\end{aligned} \tag{20}$$

Substituting (20) into equation (18), we arrive at the new approximate equation:

$$\boldsymbol{A}\boldsymbol{\rho}_{\boldsymbol{\tau}} = \frac{\sigma^2}{2}\boldsymbol{A}_{xx}\boldsymbol{\rho} + \left(r - \frac{\sigma^2}{2} - \lambda\eta\right)\boldsymbol{A}_x\boldsymbol{\rho} + (r+\lambda)\boldsymbol{A}\boldsymbol{\rho} + \lambda\boldsymbol{C}(y)\boldsymbol{\rho}. \tag{21}$$

The invertibility of $\boldsymbol{A}$ is assumed by general RBF theory; cf. for example, Buhmann [3], Powell [25] or Wendland [30]. We can therefore multiply both sides of (21) by $\boldsymbol{A}^{-1}$, which can be determined by Gaussian elimination with partial pivoting. As a result, we obtain the following homogeneous system of ODEs with constant coefficients:

$$\begin{aligned}\boldsymbol{\rho}_\tau &= \boldsymbol{A}^{-1}\Big(\frac{\sigma^2}{2}\boldsymbol{A}_{xx} + \big(r - \frac{\sigma^2}{2} - \lambda\eta\big)\boldsymbol{A}_x + (r+\lambda)\boldsymbol{A} + \lambda\boldsymbol{C}(y)\Big)\boldsymbol{\rho} \\ &\equiv \boldsymbol{\Theta}\boldsymbol{\rho} \end{aligned} \tag{22}$$

where $\boldsymbol{\Theta}$ is defined by the left hand side. After some numerical experimentation, we found that the matrix $\boldsymbol{\Theta}$ is very stiff and we therefore have to solve the ODEs by an implicit method, e.g. a modified Rosenbrock formula of order 2, the trapezoidal rule or TR-BDF2, an implicit Runge-Kutta formula with a first stage that is a trapezoidal rule step and a second stage that is a backward differentiation formula of order two. In this paper we use latter.

If we use the adaptive methodology, the matrix $\boldsymbol{A}$ becomes non-symmetrical: $\boldsymbol{A} = \big(\phi_{c_j}(|x_i - x_j|)\big)_{1\le i,j\le N}$ and similarly for $\boldsymbol{A}_x = \big(\phi'_{c_j}(|x_i - x_j|)\big)_{1\le i,j\le N}$, $\boldsymbol{A}_{xx} = \big(\phi''_{c_j}(|x_i - x_j|)\big)_{1\le i,j\le N}$ and the integral term. Invertibility of $\boldsymbol{A}$ can no longer be assumed but will have to be checked numerically.

We observe that no theoretical convergence analysis of the algorithm presented was attempted, as in most of the existing literature on numerical RBF schemes-we hope to deal with this issue in a further paper. To assess the accuracy of our RBF-algorithm, we will simply compare our numerical results with the exact solution of the Merton model in the European case, and with numerical results by other methods from the literature in the American case. This paper should be seen as an exploratory study for applying RBF methods to jump-diffusion models. Moreover, as far as the another is aware, no theoretical convergence and stability analysis of RBF-schemes exists as yet even for the Black and Scholes or for the heat equation.

4 Numerical Results

4.1 Non-smooth put payoff function

We first compare the approximation errors of the RBF-interpolation of the (non-smooth) put pay-off (4) using ESM with those using ARSM. To measure the accuracy of our RBF-approximation, we use a set of evaluation points $\hat{x}_i^{\Delta x}$, for which we will simply take the grid points

$$\hat{x}_i := \hat{x}_i^{\Delta x} = \hat{x}_{\min} + j\Delta\hat{x},\ j = 1, 2, \ldots, G. \tag{23}$$

Here $\Delta\hat{x} = (\hat{x}_{\max} - \hat{x}_{\max})/G$ with $x_{\min} \le \hat{x}_{\min} \le \hat{x}_{\max} \le x_{\max}$ and G is the number of the evaluation points chosen. We will use the following two norms for the errors, the max error:

$$E_\infty = \max_{0\le i\le G} |f(\hat{x}_i) - \widehat{u}(\hat{x}_i)|, \tag{24}$$

and the root-mean-square (rms) error:

$$E_2 = \sqrt{\frac{1}{G}\sum_{0\le i\le G} |f(\hat{x}_i) - \widehat{u}(\hat{x}_i)|^2}, \tag{25}$$

where $f(\hat{x}_i)$ is the exact value of the pay-off function (4) and $\widehat{u}(\hat{x}_i)$ is the value of its RBF-interpolant in $\hat{x}_i$.

In our numerical experiment we implement ARSM and ESM in MATHLAB R2007b. We select our maximum and minimum logarithm price $x_{\min}$ $\big(= \log(S_{\min})\big)$ and $x_{\max}$ $\big(= \log(S_{\max})\big)$ as -6 $(\log(0.002478))$and 6

(log(403.428793)) respectively. For θ_a and θ_c in ARSM, we choose 10^{-05} and 10^{-10} respectively. We set our strike, K, equal to 1 in (4).

In figure 1 and table 1, we interpolate the initial put function in (4) using ARSM and ESM. We see in figure 1 that ARSM can achieve a much better interpolation. Note that the axes in figure 1 are identically scaled for the ARSM and the ESM. We have taken the total number of evaluation points, G, equal to 3000, while the total number of interpolation points N in both the ARSM and ESM scheme was 521. The evaluation points are of course taken differently from the interpolation points. The blue line is the graph of the put pay-off (4), and the black circles represent the values of the RBF-interpolant in the interpolation points x_i; these lie all on the blue line, as they of course should (this is basically a check that the RBF-interpolation scheme is properly implemented). The red stars represent the values of the RBF-interpolant in the evaluation points $\hat{x}_i$. As can be seen from the figure, the line of red stars oscillates around the graph of (4) in case of the ESM-scheme, while there is no obvious difference between the two lines for the adaptive scheme.

In table 1, we compare both types of errors, E_∞, in (24) and, E_2, in (25) of ARSM and ESM. As seen in this table, ARSM shows both a lower E_∞ and E_2 than ESM.

	ARSM	ESM	ARSM	ESM
G	E_∞ of Put		E_2 of Put	
3000	8.8381e-006	0.0022	1.1919e-006	1.1200e-004

Table 1: *E_∞ and E_2 of the RBF approximation from an initial put function in (4) using ARSM and those using ESM. G is the number of evaluation points.*

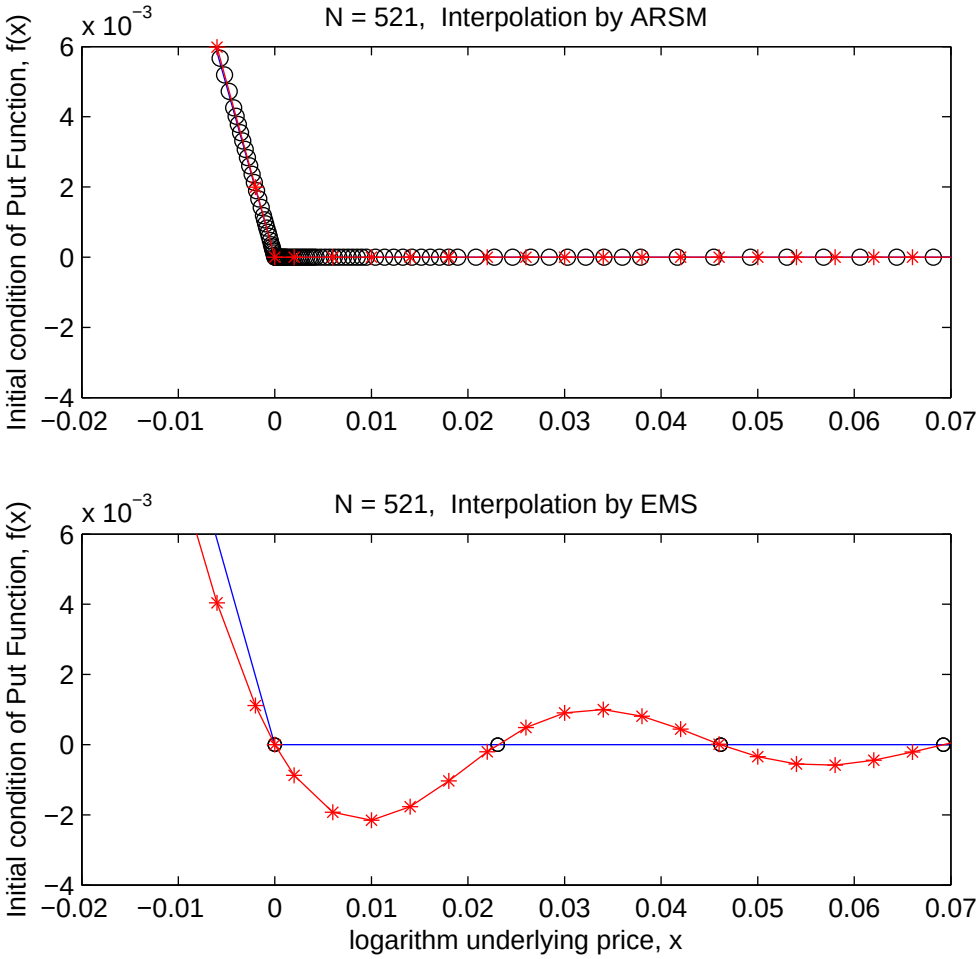

Figure 1: Graphical representation of RBF approximation of an initial put function in (4) by implementing ARSM and ESM around the strike, K=1, where a kink exists. N, the number of interpolation points, is 521. G, the number of evaluation points, is 3000. The approximate values of the RBF-interpolants outside of the interpolation points are represented by the red star line. The blue line represents graph of (4) and the black circles represent the values of the RBF-interpolant in the interpolation points.

4.2 European Option

In this section we present the numerical results of ARSM and ESM, and compare these with both Merton's analytical option price formula for puts, and with the results of Briani's finite difference algorithm in Briani *et al.* [2]. We use the same formula, (23), to define our evaluation points. We set $\hat{x}_{\min} = K - 10$ and $\hat{x}_{\max} = K + 10$ where K is a strike price. Apart from using two error measures, both (24) and (25), we define two other error measures, the absolute error:

$$E_{\text{abs.}}(x,t) = |V(e^x,t) - \widehat{u}(x,t)|, \tag{26}$$

and the relative error:

$$E_{\text{rel.}}(x,t) = \frac{|V(e^x,t) - \widehat{u}(x,t)|}{V(e^x,t)}. \tag{27}$$

where $V(e^x,t)$ and $\widehat{u}(x,t)$ are the exact value and approximate value at the point (x,t).

It is known [23] that the analytical price of a European put option in the Merton Jump-diffusion model is given by

$$V(S,t) = \sum_{m=0}^{\infty} \frac{e^{-\lambda(1+\eta)\tau}((\lambda(1+\eta)\tau)^m}{m!} V_{BS}(S,\tau,K,r_m,\sigma_m). \tag{28}$$

where $\tau = T - t$ is the time to maturity, $\eta = e^{\mu_J + \frac{\sigma_J^2}{2}} - 1$ represents the expected percentage change in the stock price originating from a jump, $\sigma_m^2 = \sigma^2 + \frac{m\sigma_J^2}{\tau}$ the observed volatility, $r_m = r - \lambda\eta + m\log(1+\eta)/\tau$ and V_{BS} the Black-Scholes price of a put, computed as

$$V_{BS}(S,\tau,K,r,\sigma) = Ke^{-r\tau}\Phi(-d_2) - S\Phi(-d_1) \tag{29}$$

where $\Phi(\cdot)$ is the cumulative normal distribution and

$$d_1 = \frac{\log(\frac{S}{K}) + (r + \frac{1}{2}\sigma^2)\tau}{\sigma\sqrt{\tau}}, \quad d_2 = d_1 - \sigma\sqrt{(\tau)}.$$

Our RBF-algorithm for numerically solving (3) with initial condition (4) runs as follows:

1. Find the RBF-approximation to the initial value $u(x,0)$ either using ESM or ARSM. This will provide us with a set of interpolation (or collocation) points $x_1,\ldots,x_n$, together with an initial vector $\boldsymbol{\rho}(0) = \big(\rho_1(0),\ldots,\rho_N(0)\big)$.
2. Then use $\boldsymbol{\rho}(0)$ as initial value for the system (22). By using any stiff ODE solver, we find out the $\boldsymbol{\rho}(T)$ at time T.
3. Finally, substitute $\boldsymbol{\rho}(T)$ back into $\sum_{j=1}^{N}\rho_j(T)\phi(|x - x_j|)$ to get an approximate value of $u(x,T)$.

In our numerical experiment we implement the algorithm in MATHLAB R2007b. We select our maximum and minimum logarithm price $x_{\min}$ $\big(\log(S_{\min})\big)$ and $x_{\max}$ $\big(\log(S_{\max})\big)$, as before, equal to -6 and 6 respectively. By experimentation, we do not need as many interpolation points if we restrict the strike price around an interval $x \in [-1,1]$; therefore, in table 2, we scale down K to K/100. Moreover, we use function *quadv* which implements recursive adaptive Simpson quadrature for computing equation (19) as well as function *ode*23*tb* which implements TR-BDF2 for the calculation of equation (22).

In order to illustrate that ARSM is better than ESM, a comparison of E_∞ and E_2 between both methods at increasing time intervals is shown in table 3. Table 3 shows that the errors in the approximate solution

computed using the adaptive ARSM-scheme stay small, even when T is close to zero, in contrast to the output of the ESM-scheme. Both for ARSM and ESM, the errors increase slowly, if at all, with increasing T, although the errors from using ARSM are much smaller.

In Table 4, we compare the results of the FDM used in Briani *et al.*'s paper [2] with those using ARSM and ESM. The ARSM-scheme can achieve lower $E_{\text{abs}}(\log S)$ than ARS-233 scheme, Explicit scheme and the ESM-scheme.

Table 5 highlights the accuracy of pricing put values of ARSM and ESM by analysing their $E_{\text{rel.}}(\log S, T)$ at different times T and for different S. Three particular put prices are examined individually, representing three different cases. Firstly, $S = 90$ represents the in-the-money case in put. The second, $S = 100$, is the at-the-money case for a put option, and in the last, $S = 110$ represents the out-of-the-money case for put options. As before, we choose the first column of the input parameter of table 2 for this experiment.

Figure 2 demonstrates that ARSM gives a smaller $E_{\text{abs}}(\log S, T)$, around 6.5×10^{-4} or below, than ESM does. The vertical axis labels "Abs. Error" represents $E_{\text{abs}}(\log S, T)$, and the latter is plotted against S and T; S ranges from 90 to 110 and T from 0 to 1. In our numerical calculations we let S increase by $\Delta S = 0.1$ and T by time-steps of $10e^{-12}$. From top to bottom, the first graph represents $E_{\text{abs}}(\log S)$ of ARSM and the next one that of ESM. As illustrated in figure 2, with ESM there is a substantial amount of oscillation along the S for small T. Also, the absolute error is highest around or at the strike and can reach almost 0.2 with the ESM-scheme. This is in line with what we found in section 4.1. The pronounced oscillation around the strike in figure 1 reduces the precision of the calculated put prices. It not only affects the quality of the approximation in the vicinity of the strike, but also in some regions further away.

Parameter Values		
σ	0.15	0.2
r	0.05	0.05
σ_J	0.45	0.8
μ_J	-0.9	0
λ	0.1	0.1
T	1e-06/0.25/1/5	1
K	100	100

Table 2: *Column 1 input parameters (except $T = 1e-06$, 1 and 5) are copied from [7] which d'Halluin, Forsyth and Vetzal used to approximate American put options and European call options in 2004. Originally these parameters come from [1] with which Andersen and Andreasen used to calculate European call options on the S&P 500 stock indies in April of 1999. Column 2, input parameters used to value European call and put options under the Merton Jump-diffusion Model. These parameters are copied from [2] which Briani, Natalini and Russo used to price European put and call options in 2007.*

		ARSM	ESM	ARSM	ESM
T	G	E_∞ of Put		E_2 of Put	
1e-06	201	7.955929e-05	2.184118e-01	9.862619e-06	8.434566e-02
0.25	201	4.029000e-04	2.277197e-02	2.614984e-04	1.769887e-02
1	201	6.423055e-04	1.047691e-02	4.422346e-04	8.169592e-03
5	201	9.843645e-04	7.308420e-01	6.116825e-04	6.854095e-01

Table 3: *E_∞ and E_2 of a European put option are presented by using input parameters provided in the first column of Table 2. G is the number of evaluation points. T is time-to-maturity.*

	Explicit scheme ([2])			$ARS-233$ scheme ([2])			
S	$Mesh$	$Value$	$E_{\text{rel.}}(\log S,T)$	$Mesh$	$Value$	$E_{\text{rel.}}(\log S,T)$	$Exact$
100	1024	8.319940	2.577970e-03	1024	8.326102	1.839249e-03	8.341436
	ARSM			ESM			
S	N	$\widehat{u}(\log S,T)$	$E_{\text{rel.}}(\log S,T)$	N	$\widehat{u}(\log S,T)$	$E_{\text{rel.}}(\log S,T)$	$Exact$
100	521	8.341670	2.804294e-05	1024	8.330652	1.293062e-03	8.341436

Table 4: *Comparative values of a European put option using ARSM and ESM versus those using FDM in Briani's methods in [2].* $E_{\text{rel.}}(\log S,T)$ *is relative error.* S *is the stock price.* N *is the number of interpolation points. Mesh is the number of mesh points. Time-to-maturity,* T*, is 1.* $\widehat{u}(\log S,T)$ *is the approximate value of a European put by using ARSM or ESM. Value is approximate value of a European put by using Explicit Scheme or ARS-233 scheme. The parameters are provided in the second column of Table 2.*

	N=521					
	ARSM			ESM		
T	S	$\widehat{u}(\log S,T)$	$E_{\text{rel.}}(\log S,T)$	$\widehat{u}(\log S,T)$	$E_{\text{rel.}}(\log S,T)$	$Exact$
1e-06	99.9	1.000009e-01	5.885266e-05	5.280614e-02	4.719125e-01	9.999504e-02
1e-06	100.0	5.904871e-03	1.329438e-02	6.849907e-05	9.885538e-01	5.984431e-03
1e-06	100.1	1.138495e-05	1.0543530	-4.661364e-02	8.412183e+03	5.541865e-06
0.25	90.0	9.285133	3.072185e-05	9.272727	1.366755e-03	9.285418
0.25	99.9	3.185076	7.507332e-05	3.185076	6.815555e-03	3.184837
0.25	100.0	3.149258	7.372928e-05	3.127410	6.864284e-03	3.149026
0.25	100.1	3.113936	7.472784e-05	3.092181	6.911995e-03	3.113703
0.25	110.0	1.400927	1.844433e-04	1.394685	4.639699e-03	1.401186
1	90.0	10.303322	6.218437e-05	10.293486	1.016785e-03	10.303963
1	95.0	8.188955	5.814502e-05	8.179676	1.191192e-03	8.189431
1	100.0	6.684307	2.016808e-05	6.676282	1.220705e-03	6.684441
1	105.0	5.653948	7.896656e-05	5.648129	1.108153e-03	5.654395
1	110.0	4.961079	7.478172e-05	4.956963	9.042944e-04	4.961450
5	90.0	15.136039	6.503026e-05	14.566859	3.766688e-02	15.137023
5	95.0	14.200279	5.289616e-05	13.548645	4.593930e-02	14.201030
5	100.0	13.370351	3.583254e-05	12.670378	5.238657e-02	13.370830
5	105.0	12.626235	3.278034e-05	11.902553	5.734664e-02	12.626649
5	110.0	11.952952	2.315001e-05	11.222387	6.114181e-02	11.953229

Table 5: *Comparison between analytical values of a European put option with its approximate values by using ARSM and ESM.* T *is the time-to-maturity.* $\widehat{u}(\log S,T)$ *is the approximate value.* $E_{\text{rel.}}(\log S,T)$ *is relative error. Exact is the analytical value of a European put option in (28).* N *is the number of interpolation points. The parameters are provided in the first column of Table 2.*

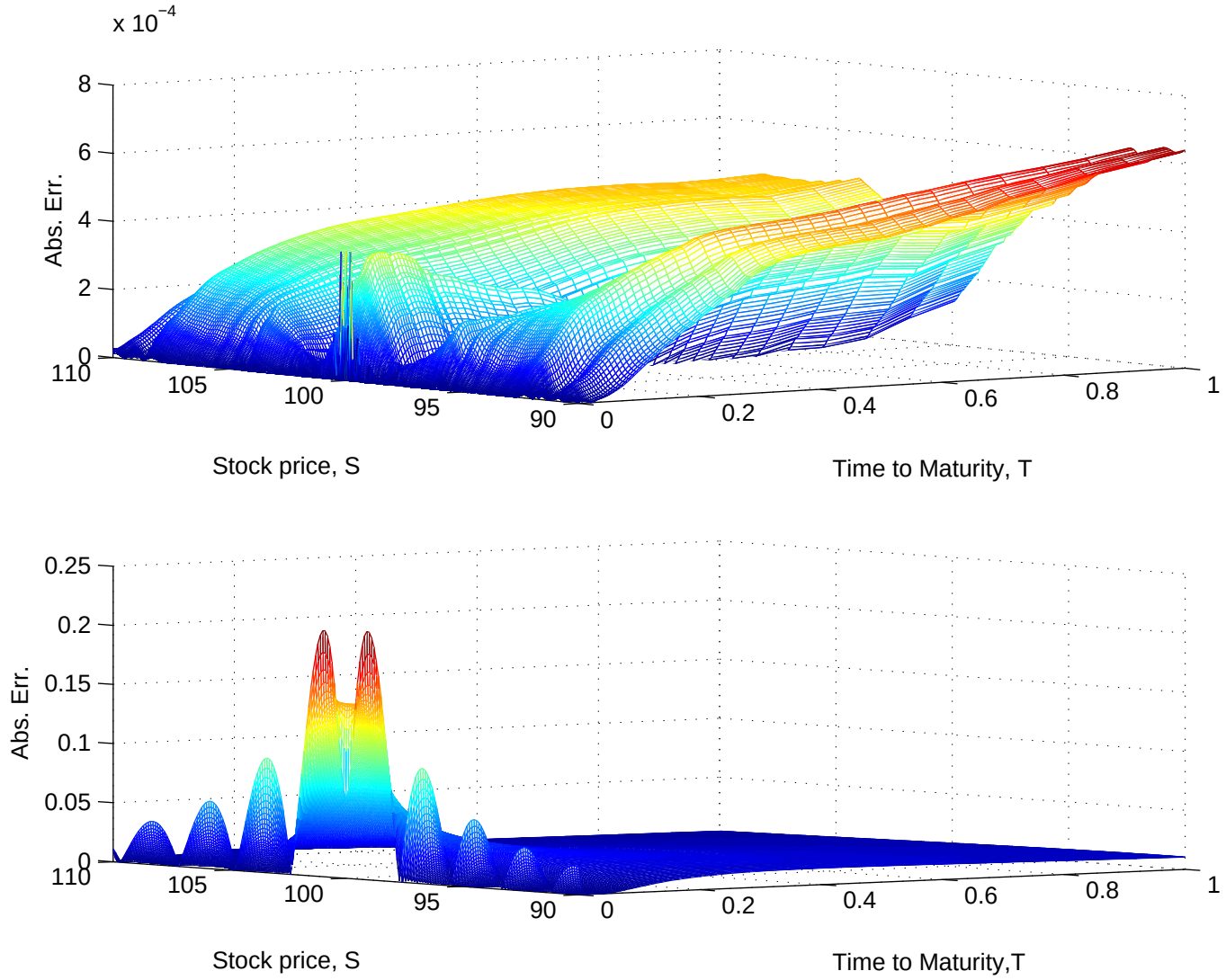

Figure 2: Graphical representation of absolute error of a European put between ARSM-RBF (the top graph) and ESM-RBF (the bottom graph) approximate values and the analytical values in (28). *Abs. Err.* is $E_{\text{abs}}(\log S, T)$. T is the time-to-Maturity . S is the underlying stock price. The input parameters are provided in the first column of Table 2.

4.3 American Put Options

In this section we adapt our RBF-algorithm to compute American put-option prices. We then compare the option prices obtained from our RBF-algorithm, ARSM and ESM with those of Forysth *et al.*'s FDM in [7]. As mentioned in section 2, an American put option problem is a free boundary problem because of the possibility of early exercise at any point during its life, leading to the free boundary condition:

$$u(x,\tau) = \max\big(K - e^x, u(x,\tau)\big).$$

Together with the smooth pasting condition mentioned in section 2, this uniquely determines the exercise boundary.

As before, we use both ARSM and ESM to approximate $u(x,0) = \max(K - e^x, 0)$ and then continue to work with the interpolation points found at $\tau = 0$. The algorithm now reads as follows:

1. Divide time-to-maturity T by total numbers of time-steps M to obtain time interval Δt and create a list of equally spaced time-points $m\Delta t$, $m \in \{0, 1, 2, \ldots, M-1\}$.
2. Find the RBF-approximation to the initial value $u(x,0)$ either using ESM or ARSM. This will provide us with a set of collocation or interpolation points $x_1, \ldots, x_n$, together with an initial vector $\boldsymbol{\rho}(0) = \big(\rho_1(0), \ldots, \rho_N(0)\big)$.
3. Assume we have already determined $\boldsymbol{\rho}(m\Delta t)$ (if $m = 0$, we have $\boldsymbol{\rho}(0)$) in equation (22). Solve the system of (stiff) ODEs to find $\boldsymbol{\rho}\big((m+1)\Delta t\big)$ at the next successive time-step, $(m+1)\Delta t$.
4. Then at time $(m+1)\Delta t$, for each interpolation point x_i, define

$$u\big(x_i, (m+1)\Delta t\big) = \max\big((K - e^{x_i}), \sum_{j=1}^{N} \rho_j\big((m+1)\Delta t\big)\phi(|x_i - x_j|)\big).$$

5. Find a new vector $\boldsymbol{\rho}\big((m+1)\Delta t\big)$ such that $u\big(x_i, (m+1)\Delta t\big) = \sum_{j=1}^{N} \rho_j\big((m+1)\Delta t\big)\phi(|x_i - x_j|)$ for all i.
6. Repeat Step 3.) to 5.) until $m = M - 1$.
7. Finally, substitute $\boldsymbol{\rho}(T)$ back into $\sum_{j=1}^{N} \rho_j(T)\phi(|x - x_j|)$ to get an approximate value of $u(x,T)$.

The settings of our numerical experiment are the same as those in section 4.2. In table 6, $\Delta\widehat{u}$ designates the difference between the values obtained for $\widehat{u}(\log S, T)$ for two successive values of the number of time-steps, M (listed in the first column of table 6); $\Delta\widehat{u}$ decreases with increasing number of time-steps. This indicates numerically, at least, that our method converges. In table 7, we use Forsyth *et al.*'s FDM in [7] as an indicator of the accuracy of our American option prices. Forysth *et al.* obtain their best results using a scenario involving 4065 mesh points and 940 time-steps. Taking their results as a benchmark for the prices computed using our ESM and ARSM schemes, but with only 521 interpolation points, we see that for the three values of S considered, ARSM gives a much smaller absolute error, $E_{\text{abs.}}(\log S, T)$, especially in the at-the-money and out-of-the-money cases.

	$S = 90$		$S = 100$		$S = 110$	
M	$\widehat{u}(\log S, T)$	$\Delta\widehat{u}$	$\widehat{u}(\log S, T)$	$\Delta\widehat{u}$	$\widehat{u}(\log S, T)$	$\Delta\widehat{u}$
25	9.999991	N.A.	3.236244	N.A.	1.418371	N.A.
50	10.000703	7.120572e-06	3.238813	2.569763e-05	1.418952	5.814587e-06
100	10.000968	2.648464e-06	3.240127	1.313583e-05	1.419255	3.030001e-06
200	10.001030	6.265351e-07	3.240788	6.608718e-06	1.419411	1.552927e-06
400	10.001597	5.666263e-06	3.241118	3.306593e-06	1.419489	7.876438e-07
800	10.002106	5.093106e-06	3.241283	1.647557e-06	1.419529	3.963116e-07
1600	10.002388	2.813500e-06	3.241365	8.190645e-07	1.419549	1.983668e-07

Table 6: *Value of an American put option by using ARSM. Number of interpolation points is 521. M is the number of time-steps. $\widehat{u}(\log S, T)$ is the approximate value of an American option by using ARSM. $\Delta\widehat{u}$ is the change from one level of refinement to the next. Time-to-maturity, T, is 0.25. The parameters are provided in the first column of Table 2.*

		$S = 90$		$S = 100$		$S = 110$	
		ARSM					
N	M	$\widehat{u}(\log S, T)$	$E_{\text{abs.}}(\log S, T)$	$\widehat{u}(\log S, T)$	$E_{\text{abs.}}(\log S, T)$	$\widehat{u}(\log S, T)$	$E_{\text{abs.}}(\log S, T)$
521	940	10.002189	0.001633	3.241308	5.679744e-05	1.419535	2.679948e-04
		ESM					
N	M	$\widehat{u}(\log S, T)$	$E_{\text{abs.}}(\log S, T)$	$\widehat{u}(\log S, T)$	$E_{\text{abs.}}(\log S, T)$	$\widehat{u}(\log S, T)$	$E_{\text{abs.}}(\log S, T)$
521	940	10.00000	0.003822	3.225347	0.015904	1.417204	0.002599
		Rannacher timestepping with variable timestep sizes					
$Mesh$	M	Value	$E_{\text{abs.}}(\log S, T)$	Value	$E_{\text{abs.}}(\log S, T)$	Value	$E_{\text{abs.}}(\log S, T)$
4065	940	10.003822	N.A.	3.241251	N.A.	1.419803	N.A.

Table 7: *Comparison of values of an American put option using ARSM and ESM against those in Forysth et al. [7]. N is the number of interpolation points. M is the number of time-steps. $Mesh$ is the number of mesh points. $Value$ is the approximate value of an American option by using Rannacher timestepping with variable timestep sizes. $\widehat{u}(\log S, T)$ is the approximate value of an American option by using ARSM or ESM. $E_{\text{abs.}}(\log S, T)$ is the absolute value of the difference in the point S of our RBF-solution with that of Forysth et al. [7]. Time-to-maturity, T, is 0.25. The parameters are provided in the first column of Table 2.*

5 Conclusions

We have implemented a RBF method to solve the PIDE boundary value problem for pricing American and European put options on a non-dividend-paying stock in a Merton jump-diffusion market [23]. We also compared ARSM and ESM for determining RBF-interpolation points. Our results suggest that one can achieve a high accuracy by implementing ARSM which involves a limited number of interpolation points. Moreover, several drawbacks associated with grid-based methods like the FDM have been avoided: we do not have to make assumptions on the behavior of the solutions outside of the solution domain, we seem to avoid the stability problems associated with explicit or implicit finite difference schemes. Moreover, we dramatically improve the accuracy of pricing put options in particular for small times to maturity, by implementing ARSM.

Our Method extends in principle to pure jump Lévy type models for the underlying stocks, like Variance Gamma (VG) or CGMY.

References

[1] Andersen, L. and Andreasen, J., 2000. Jump-diffusion Processes: Volalitility Smile fitting and Numerical Methods for Option Pricing. *Review of Derivatives Research*, 4, pp. 231 - 262.

[2] Briani, M., Natalini R. and Russo, G., 2007. Implicit-explict Numerical Schemes for Jump-diffusion Processes. *Calcolo*, 44, pp. 33 - 57.

[3] Buhmann. M. D., 2003. *Radial basis functions: theory and implementations*, volume 12 of Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, Cambridge.

[4] Carr, P. Geman, H. Madan, D. B. and Yor. M., 2002. The Fine Structure of Asset Returns: an Empirical Investigation. *Journal of Business*, 75, pp. 305 - 332.

[5] Cont, R. and Voltchkova, E., 2005. A Finite Difference Scheme for Option Pricing in Jump Diffusion and Exponential Lèvy Models. *SIAM Journal on Numerical Analysis*, 43, pp. 1596 - 1626.

[6] Cont, R. and Tankov, P., 2004. *Financial Modelling With Jump Processes*, CHAPMAN & HALL/CRC Finanical Mathematics Series.

[7] d'Halluin, Y., Forsyth, P.A. and Vetzalz, K.R., 2005. Robust Numerical Methods for Contingent Claims under Jump Duffusion Process. *IMA J. Num. Anal.*, 25, pp. 87 - 112.

[8] d'Halluin, Y., Forsyth, P.A. and Labahn, G., 2004. A Penalty Method for American Options with Jump Diffusion Processes. *Numerische Mathematik*, 97(2), pp. 321 - 352.

[9] Driscoll, T. A. and Heryudono, A., 2007. Adaptive Residual Subsampling Methods for Radial Basis Function Interpolation and Collocation Problems. *Computers Math. Appl.*, 53, pp. 927 - 939.

[10] Fasshauer, G. E., Khaliq, A. Q. M. and Voss, D. A., July, 2004. A Parallel Time Stepping Approach Using Meshfree Approximations for Pricing Options with Non-smooth Payoffs. *Proceedings of Third World Congress of the Bachelier Finance Society*, Chicago.

[11] Fasshauer, G. E. and Zhang, J. G.. On Choosing Optimal Shape Parameters for RBF Approximation. *To be appeared in Numerical Algorithms.*

[12] Fasshauer, G. E., Khaliq, A. Q. M. and Voss, D. A., 2004. Using Meshfree Approximation for Multi-Asset American Option Problems. *J. Chinese Institute Engineers*, 27(4), pp. 563 - 571.

[13] Fornberg, B. and Wright, G., 2004. Stable Computation of Multiquadric Interpolants for All Values of the Shape Parameter. *Comput. Math. Appl.*, 47, pp. 497 - 523.

[14] Hon, Y.C. and Mao, X. Z., A Radial Basis Function Method for Solving Options Pricing Model. *Financial Engineering*, to appear.

[15] Hull, J.C., 2007. *Options, Futures, and Other Derivatives, Seventh Edition*, Pearson Education.

[16] Johnson, C., 1987. Numerical Solutions of Partial Differential Equations by the Finite Element Method. *Cambridge University Press*, Cambridge.

[17] Kansa, E. J. and Carlson, R. E., 1992. Improved Accuracy of Multiquadric Interpolation Using Variable Shape Parameters. *Comput. Math. Applic.*, 24, pp. 99 - 120.

[18] Kansa, E. J. and Carlson, R. E., 1994. The Laplace Transofrm Multiquadrics Method: A Highly Accurate Scheme For The Numerical Solution of Linear Partial Differential Equations. *Journal of Applied Science & Computations*, 1(2), pp. 375 - 407.

[19] Kansa, E.J., 1990. Multiquadrics - A Scattered Data Approximation Scheme with Applications to Computational Fluid Dynamics - I. Surface approximations and partial derivatives estimates *Comput. Math. Appl.*, 19(8/9), pp. 127 - 145.

[20] Kansa, E. J., 1990. Multiquadrics-A Scattered Data Approximation Scheme with Applications to Computational Fluid-Dynamics-II: Solutions to Parabolic, Hyperbolic and Elliptic Partial Differential Equations. *Comput. Math. Appl.*, 19(8/9), pp. 147 - 161.

[21] Madan, D.B. and Seneta, E., 1990. The Variance Gamma (V.G.) Model for Share Market Returns. *Journal of Business*, 63, pp. 511 - 524.

[22] Matache, A., Schwab, C. Wihler, T. P., 2005. Fast Numerical Solution of Parabolic Integrodifferential Equations with Applications in Finance. *SIAM Journal on Scientific Computing* , 27(2), pp. 369 - 393.

[23] Merton, R. C., 1976. Option Pricing when Underlying Stock Returns are Discontinuous. *Journal of Financial Economics*, 3, pp. 125 - 144.

[24] Pham, H., 1997. Optimal Stopping, Free Boundary, and American Option in a Jump-diffusion Model. *Appl. Math. Optim.*, 35, pp. 125 - 144.

[25] Powell, M. J. D., 1992. The Theory of Radial Basis Function Approximation in 1990, in: W. Light (Ed.). *Advances in Numerical Analysis, II: Wavelets, Subdivision Algorithms and Radial Basis Functions*, Clarendon Press, Oxford, pp. 105 - 210.

[26] Rannacher, R., 1984. Finite Element Solution of Diffusion Problems with Irregular Data. *Numerische Mathematik*, 43, pp. 309 - 327.

[27] Sarra, S. A., 2005. Adaptive Radial Basis Function Methods for Time Dependent Partial Differential Equations. *Applied Numerical Mathematics*, 54(1), pp. 79 - 94.

[28] Schoutens, W., 2006. Exotic Options under Lévy Models: an Overview. *Journal of Computational and Applied Mathematics*, 189, pp. 526 - 538.

[29] Shampine, L. F., 2008. Vectorized Adaptive Quadrature in MATLAB. *Journal of Computational and Applied Mathematics*, 211(2), pp. 131 - 140.

[30] Wendland, H., 2005. *Scattered Data Approximation*, volume 17 of Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, Cambridge.